CHAPTER ONE: INTRODUCTION

1.1 Background of the Study

The stock market is a highly dynamic environment where prices are influenced by a wide range of factors, including historical trends, economic indicators, and investor sentiment. Predicting stock prices has long been a complex challenge due to the inherent volatility of financial markets and the unpredictable nature of external events. Traditional forecasting methods, although valuable, often fall short when it comes to analyzing large volumes of data or identifying subtle and complex patterns in market behavior.

In recent years, advancements in artificial intelligence (AI) and machine learning (ML) have introduced new opportunities for improving stock price prediction. Machine learning models such as Prophet offer powerful tools for analyzing historical stock data and forecasting future trends.

This study focuses on the development of a web-based stock price prediction system that utilizes machine learning techniques, particularly the Prophet model, to assist investors and financial analysts. The goal is to provide a user-friendly platform for making data-driven investment decisions.

1.2 Problem Statement

Investors and financial analysts often struggle to make informed investment decisions due to the unpredictable nature of stock prices. Traditional methods of analysis, such as interpreting financial reports, relying on expert opinions, or using technical indicators, provide some insight but may fail to uncover deeper patterns that drive price movements. Furthermore, access to reliable and intuitive prediction tools remains limited for many investors, particularly those without technical expertise.

This study aims to address these challenges by developing a web-based stock price prediction system that leverages machine learning. By analyzing historical stock data and employing predictive models, the system is designed to enhance the accuracy of forecasts and help users make better-informed decisions, thereby reducing investment risks.

1.3 Aim of the Project

The aim of this project is to develop a web-based stock price prediction system that utilizes the Prophet machine learning model to assist investors and financial analysts in making data-driven decisions. This system will help in predicting future stock prices based on historical data, ultimately providing a more accurate and accessible forecasting tool.

1.4 Specific Project Objectives

The specific objectives of this project include:

1. To develop a web application for stock prices prediction.
2. To incorporate the web application with the Prophet Machine Learning Model to forecast future stock prices.
3. To provide interactive visualizations of stock trends, forecasts and components using Plotly.

1.5 Scope of the Project

This research focuses on developing a web-based stock price prediction system with the following features:

- Forecasting stock prices for selected companies using historical data.
- Implementing the Prophet model for time-series forecasting.
- Retrieving stock data from Yahoo Finance via the yfinance library.
- Visualizing stock data using interactive charts, including time-series and candlestick plots.
- Creating a user-friendly interface using Streamlit for seamless interaction with the system. The project will be limited to the selected stock symbols (e.g., AngloGold Ashanti Limited, MTN Group, etc.) and will not cover all available stocks in the market.

1.6 Project Limitations

The limitations of the project are as follows:

- The prediction system will only be able to forecast stock prices for the selected companies, limiting its generalizability to other stocks.
- The accuracy of the predictions depends on the quality of historical data, which may not account for all real-world factors such as sudden market events or changes in economic conditions.
- The model will only consider historical price data and will not include other factors such as company earnings reports, political events, or broader economic indicators, which can also impact stock prices.
- As the system relies on the Prophet model, its performance might be limited when compared to more complex machine learning algorithms, especially in volatile market conditions.

1.7 Academic and Practical Relevance of the Project

This study is academically relevant as it explores the intersection of machine learning and financial forecasting, a rapidly growing area in the field of financial technology (FinTech). The use of predictive models like Prophet opens up new avenues for understanding and forecasting market behavior using data-driven approaches.

Practically, the project holds significant value for investors, financial analysts, and other stakeholders in the financial market. The development of a user-friendly web application that can predict stock prices will empower users with better decision-making tools, helping them mitigate investment risks and improve their financial outcomes.

1.8 Beneficiaries of the Project

The primary beneficiaries of this project include:

- Investors: Both individual and institutional investors can use the stock price prediction system to make informed investment decisions based on historical data and machine learning forecasts.
- Financial Analysts: Analysts can leverage the predictions from the system to validate their own insights and enhance their research.
- Financial Institutions: Banks, investment firms, and advisory services can use the system to provide enhanced forecasting tools for their clients.
- Researchers: Academics studying the application of machine learning in finance may find the findings useful for further research in predictive modeling and financial analytics.

1.9 Project Activity Planning

The project will be carried out in the following phases:

1. Planning and Requirements Gathering: In this phase, the scope, objectives, and functionalities of the stock price prediction system will be clearly defined. The project plan will also include timelines and resource allocation.
2. Data Collection and Preprocessing: Stock data will be collected from Yahoo Finance using the yfinance library. The data will then be preprocessed to ensure it is suitable for machine learning modeling.
3. Model Development and Training: The Prophet machine learning model will be trained using the historical stock data to make future predictions.
4. Web Application Development: The user interface will be designed and implemented using the Streamlit framework to make it accessible and easy to use.
5. System Testing and Evaluation: The system will undergo rigorous testing to ensure it

performs as expected. This will include evaluating the prediction accuracy and user interface usability.

6. Deployment and Final Report: Once the system is ready, it will be deployed for use. The final report will be written to document the project process, results, and conclusions.

1.10 Definitions and Explanations of Terms

- Stock Market: A marketplace where stocks (shares of companies) are bought and sold.
- Stock Price Prediction: The process of forecasting the future price of stocks based on historical data and various analytical methods.
- Machine Learning: A type of artificial intelligence that enables systems to learn from data and improve their performance without being explicitly programmed.
- Prophet Model: A forecasting tool developed by Facebook that uses time-series data for predicting future trends in the data.
- Streamlit: An open-source framework used for creating data-driven web applications with minimal coding effort.

1.11 Structure of the Report

This report is organized into the following chapters:

- Chapter One: Introduction - Provides the background, problem statement, objectives, scope, significance, and organization of the study.
- Chapter Two: Literature Review - Reviews relevant literature on stock prediction models and machine learning techniques used in financial forecasting.
- Chapter Three: Methodology - Outlines the research methodology, including data collection, model selection, and system development.
- Chapter Four: Results and Analysis - Presents the results of the stock price prediction system, including model performance and user feedback.
- Chapter Five: Conclusion - Summarizes the findings of the study and offers recommendations for future work.

CHAPTER TWO: LITERATURE REVIEW

Review of System 1: Stock Price Prediction Using Deep Learning (LSTM)

1. DESCRIPTION OF SYSTEM

- Overview of the system.

  The LSTM-based stock price prediction system utilizes deep learning techniques to predict stock prices. LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) designed for time-series forecasting. The system learns from historical stock prices and other relevant data to predict future prices. LSTM is particularly well-suited for sequential data, which is essential for modeling stock price movements over time (Hochreiter & Schmidhuber, 1997).

- Models of the system
  - Context Model: The system operates in the context of stock market forecasting, taking historical stock prices and external economic indicators as inputs. The system outputs predicted stock prices for future time steps (Bao et al., 2017).
  - Interaction Model (Sequence Diagram): The sequence diagram shows the flow of data between the system and its components, including data preprocessing, model training, prediction generation, and output delivery to the user interface (Selvin et al., 2017).
  - Structural Model: The system consists of several layers: a data collection module, a data preprocessing module, the LSTM model, and the prediction and visualization interface (Fischer & Krauss, 2018).
  - Behavioral Model: The system continuously collects new stock data, trains the LSTM model, and updates predictions. When a user inputs a stock symbol, the model is invoked to generate future price forecasts (Bao et al., 2017).
- Features of the system
  - Real-time stock data collection and preprocessing
  - LSTM model for predicting future stock prices
  - Visual presentation of stock price trends
  - Predictive accuracy measurement and model evaluation

- Development tools and development environment of the system
    - Programming Languages: Python
    - Libraries: Keras, TensorFlow, Pandas, Matplotlib
    - Development Environment: Jupyter Notebook, Anaconda

2. REVIEW OF THE GOOD FEATURES

- High Accuracy: LSTM networks are highly effective for sequential data and can capture long-term dependencies in stock market data (Hochreiter & Schmidhuber, 1997).
- Flexibility: The model can be adjusted to forecast prices for different time periods (e.g., hourly, daily, or weekly) (Fischer & Krauss, 2018).
- Adaptability: The LSTM model can be fine-tuned with additional market data such as financial indicators, sentiment analysis, and news trends (Bao et al., 2017).

3. REVIEW OF THE BAD FEATURES

- Computationally Expensive: Training LSTM models requires significant computational power and time, especially when using large datasets (Fischer & Krauss, 2018).
- Data Sensitivity: LSTM models are sensitive to data noise and require high-quality, cleaned data for accurate predictions (Selvin et al., 2017).
- Overfitting Risk: The model can overfit to historical data, especially if the training dataset is not diverse enough (Bao et al., 2017).

4. SUMMARY OF THE SYSTEM REVIEW

The LSTM-based stock price prediction system is a powerful tool for financial forecasting due to its ability to model complex, sequential patterns. However, it is computationally expensive and sensitive to noisy data, which can affect prediction accuracy. Despite these limitations, the system's adaptability and high accuracy make it an attractive option for stock price prediction (Hochreiter & Schmidhuber, 1997; Fischer & Krauss, 2018).

Review of System 2: Stock Price Forecasting Using ARIMA

1. DESCRIPTION OF SYSTEM

- Overview of the system.

  The ARIMA (Autoregressive Integrated Moving Average) model is a statistical approach

used for time-series forecasting. It is often employed for stock price prediction by leveraging historical data. ARIMA works by analyzing the past values of stock prices and identifying trends, seasonal patterns, and irregularities to predict future prices (Box et al., 2015).

- Models of the system
    - Context Model: The system uses historical stock prices to model future price movements. The context involves time-series forecasting, which predicts future stock prices based on observed data (Ariyo et al., 2014).
    - Interaction Model (Sequence Diagram): The sequence diagram shows the flow of historical stock price data being fed into the ARIMA model, where it undergoes preprocessing and is used to generate a forecast (Box et al., 2015).
    - Structural Model: The system has modules for data collection, preprocessing, ARIMA model training, and prediction generation (Asteriou & Hall, 2015).
    - Behavioral Model: The system identifies patterns in past data, uses them for forecasting, and updates the model with the latest available data to improve prediction accuracy (Ariyo et al., 2014).
- Features of the system
    - Simple to implement and understand
    - Effective for stationary data with linear trends
    - Incorporates past price data for forecasting future prices
- Development tools and development environment of the system
    - Programming Languages: Python
    - Libraries: Statsmodels, Pandas, Numpy
    - Development Environment: Jupyter Notebook, Anaconda

2. REVIEW OF THE GOOD FEATURES

- Simplicity: ARIMA is relatively easy to implement and does not require a large amount of data compared to deep learning models (Box et al., 2015).

- Effectiveness with Stationary Data: ARIMA works well with time-series data that is stationary, making it a good choice for stable markets with clear trends (Asteriou & Hall, 2015).

3. REVIEW OF THE BAD FEATURES

- Poor Performance with Volatile Markets: ARIMA struggles to predict in highly volatile markets, as it assumes stationarity in the data, which is often not the case in stock markets (Ariyo et al., 2014).
- Limited to Linear Trends: ARIMA is less effective when dealing with non-linear patterns or sudden market changes (Box et al., 2015).
- Data Preprocessing Requirement: ARIMA models require thorough preprocessing and transformation of data to ensure it is stationary (Asteriou & Hall, 2015).

4. SUMMARY OF THE SYSTEM REVIEW

The ARIMA model is a well-established method for stock price forecasting, especially for data with clear trends. However, it has limitations in handling volatile and non-linear data, making it less suitable for unpredictable stock market behavior. Despite these drawbacks, ARIMA remains a useful tool for specific types of time-series data (Box et al., 2015; Ariyo et al., 2014).

Review of System 3: Facebook Prophet for Time-Series Forecasting

1. DESCRIPTION OF SYSTEM

- Overview of the system.

  Facebook Prophet is an open-source forecasting tool developed by Facebook to handle time-series data. It is particularly useful for forecasting data that exhibits seasonal trends and irregular patterns, such as stock prices. Prophet works by decomposing time-series data into three components: trend, seasonality, and holidays/events. It is known for its ease of use, robustness to outliers, and ability to handle missing data (Taylor & Letham, 2018).

- Models of the system

    - Context Model: The system takes historical stock prices and other external factors (e.g., holidays or special events) as input to forecast future prices (Taylor & Letham, 2018).
    - Interaction Model (Sequence Diagram): The sequence diagram illustrates how historical stock price data and events are input into Prophet, which generates predictions (Sean et al., 2016).
    - Structural Model: The system includes modules for data preprocessing, model training, and prediction visualization (Taylor & Letham, 2018).
    - Behavioral Model: Prophet decomposes the time-series data into components, trains the model, and updates forecasts based on new data input (Sean et al., 2016).
- Features of the system
    - Robust to missing data and outliers
    - Decomposes data into trend, seasonality, and holiday effects
    - Provides uncertainty intervals for predictions
    - Easy-to-use interface for non-technical users
- Development tools and development environment of the system
    - Programming Languages: Python, R
    - Libraries: Prophet, Pandas, Matplotlib
    - Development Environment: Jupyter Notebook, Anaconda

2. REVIEW OF THE GOOD FEATURES

- Ease of Use: Prophet is designed to be user-friendly, even for non-experts, and requires minimal tuning (Taylor & Letham, 2018).
- Robustness: It handles missing data and outliers well, making it suitable for real-world financial data (Sean et al., 2016).
- Accuracy: Prophet is known for its accuracy in forecasting stock prices, especially when the data exhibits seasonal behavior (Taylor & Letham, 2018).

3. REVIEW OF THE BAD FEATURES

- Limited for High-Frequency Data: Prophet performs better with daily or monthly data rather than high-frequency minute-level stock prices (Taylor & Letham, 2018).

- Assumption of Similar Trends: The model assumes that future trends will resemble historical patterns, which may not always hold in rapidly changing markets (Sean et al., 2016).

4. SUMMARY OF THE SYSTEM REVIEW

Facebook Prophet offers a robust and easy-to-use solution for time-series forecasting, particularly in cases where data exhibits clear seasonal patterns. However, its reliance on historical trends for future predictions and its limitations with high-frequency data could hinder its application in certain stock prediction scenarios. Despite these drawbacks, Prophet remains a powerful tool for many forecasting tasks (Taylor & Letham, 2018; Sean et al., 2016).

CHAPTER THREE

3.1 Methodology

3.1.1 Introduction

This chapter outlines the methodology adopted in the development of the Stock Price Prediction Web Application. The methodology encompasses the chosen software development model, justification for its use, a breakdown of the activities involved in each phase, and tools used throughout the development process.

3.1.2 Chosen Software Development Process Model: Agile Methodology

3.1.3 Justification for Choosing Agile Methodology

The Agile development methodology was selected due to the iterative and flexible nature of the project. The stock prediction system involves several components including data retrieval, machine learning-based prediction, and a web interface all of which benefit from regular feedback, testing, and improvement. Agile facilitates rapid prototyping, frequent evaluation of working software, and adaptive planning which are ideal for this project.

3.1.4 Outline of Agile Process Model

Agile software development is an iterative and incremental approach to software delivery. It emphasizes collaboration, flexibility, and customer feedback. The core phases of Agile used in this project are:

1. Requirement Gathering & Planning
2. Design
3. Implementation
4. Testing
5. Deployment
6. Evaluation & Feedback

3.1.5 Activities Embarked on in Each Phase

- Requirement Gathering & Planning:
  - Identified the target users (investors, finance students, data analysts).
  - Defined project goals and scope: forecasting stock prices using machine learning.
  - Selected companies and features (date selection, stock symbol, and visualization).

- Chose tools and technologies (Python, Streamlit, Prophet, yfinance, Plotly).

- Design:
  - Designed wireframes for the user interface.
  - Modeled system architecture and interactions using UML diagrams.
  - Defined user input/output flow and interaction design.
- Implementation:
  - Implemented backend logic using Python and the Prophet forecasting model.
  - Retrieved data using yfinance and processed it into suitable format.
  - Built frontend using Streamlit for real-time user interaction.
  - Integrated Plotly for data visualization.
- Testing:
  - Unit tested each component for correctness.
  - Validated the predictions generated by the Prophet model.
  - Conducted UI/UX testing to ensure usability.
- Deployment:
  - Hosted the Streamlit application locally and on the cloud for broader access.
  - Made use of GitHub for version control and project management.
- Evaluation & Feedback:
  - Gathered feedback from potential users.
  - Iteratively improved the app based on feedback (e.g., simplified date range selection, added more visualizations).

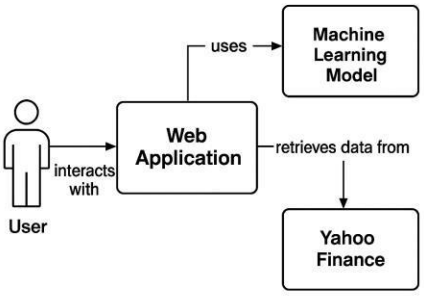
3.2 Proposed System

3.2.1 Introduction of Proposed System

The proposed system is a web-based application designed to allow users to predict and visualize stock prices. The system enables selection of predefined companies, forecasting based on a user-defined date range, and displays historical as well as predicted data in interactive charts.

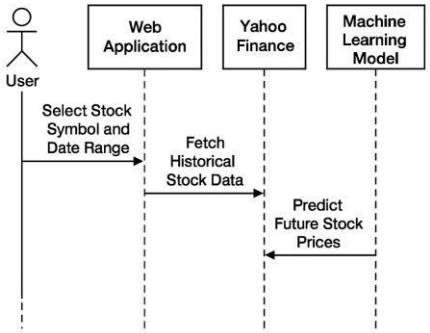
3.2.2 Models of the Proposed System

1. Context Model:



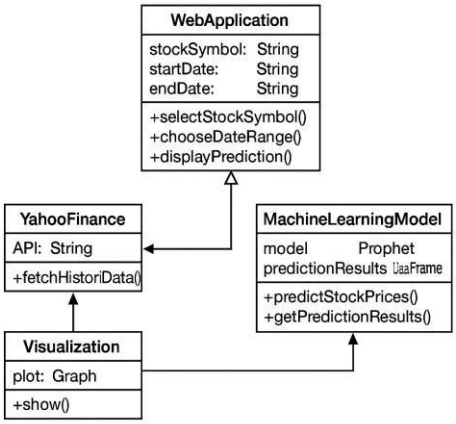
Context diagram

2. Interaction Model (Sequence Diagram):



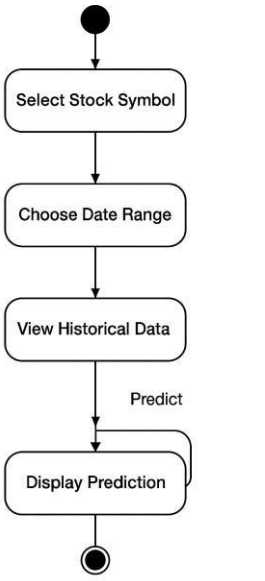
Stock Price Prediction Sequence Diagram

3. Structural Model (Class Diagram):



Structural Model of Stock Price Prediction Web Application

4. Behavioral Model (Activity Diagram):



Behavioral Model of Stock Price Prediction Web Application

3.2.3 New Features of the Proposed System

- Stock selection from a predefined list.
- Customizable date range for forecasting.
- Interactive plots (candlestick, forecast trends).
- Multiple company support.
- Prediction horizon up to 5 years.

3.2.4 Development Tools and Environment

➢ Development Tools:

- Python: For logic, data handling, and model integration.
- Streamlit: For rapid UI development and deployment.
- Prophet: For time-series forecasting.
- yfinance: For fetching historical stock data.
- Plotly: For interactive data visualization.

➢ Development Environment:

- IDE: Visual Studio Code
- Operating System: Windows 10

- Version Control: Git & GitHub
- Browser: Google Chrome

3.2.5 Review of the Good Features

- User-friendly interface with real-time interaction.
- Reliable data sourcing using yfinance.
- Accurate forecasts using Prophet.
- Insightful and interactive visualizations with Plotly.
- Scalable design supporting multiple companies and date ranges.

3.2.6 Review of the Bad Features

- Dependency on internet connectivity for data retrieval.
- Limited to only selected companies, limiting its generalizability to other stocks.
- Model accuracy may decrease during highly volatile market conditions.

3.2.7 Summary of the System Review

The Stock Price Prediction Web Application successfully meets its objectives by integrating reliable forecasting models with an intuitive user interface. It provides users with an easy-to-use tool to analyze and predict stock market trends, offering educational and investment benefits. While there are areas for enhancement, especially around model accuracy and offline capabilities, the system as developed is a robust solution for time-series forecasting in finance.

CHAPTER FOUR

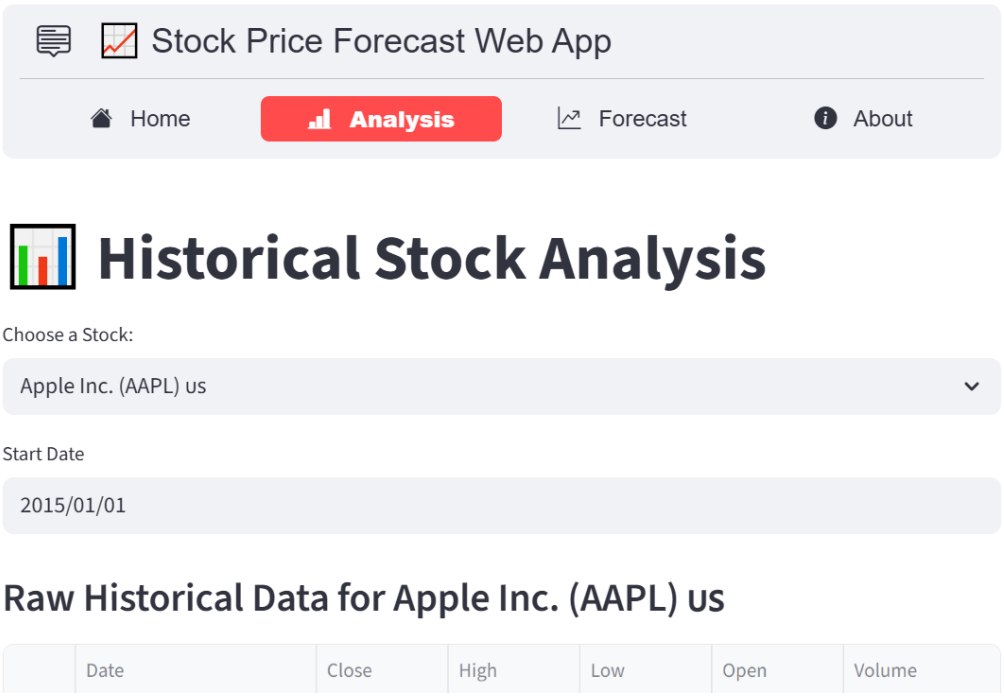
4.1 Introduction

This chapter presents the implementation, testing, and results of the Stock Price Prediction Web Application. It details how the logical design was mapped onto a physical platform, the development and integration of individual modules, and the testing procedures undertaken to verify, validate, and secure the system. The chapter concludes with the outcomes of the testing phase, including feedback and corresponding modifications.

4.2 Mapping Logical Design onto the Physical Platform (System Development)

The logical design developed in Chapter Three was translated into a functional system using Python and Streamlit as the core technologies. The application modules were developed using modular programming techniques to separate concerns between data handling, model prediction, and user interaction. All development was conducted in Visual Studio Code on a Windows 10 environment, with version control managed via GitHub. The entire system was then deployed to a Streamlit sharing platform for user access and feedback.

4.3 System Modules Implementation (UI, DB, etc.)



Raw Historical Data for MTN Group Limited (MTNOY) ZA

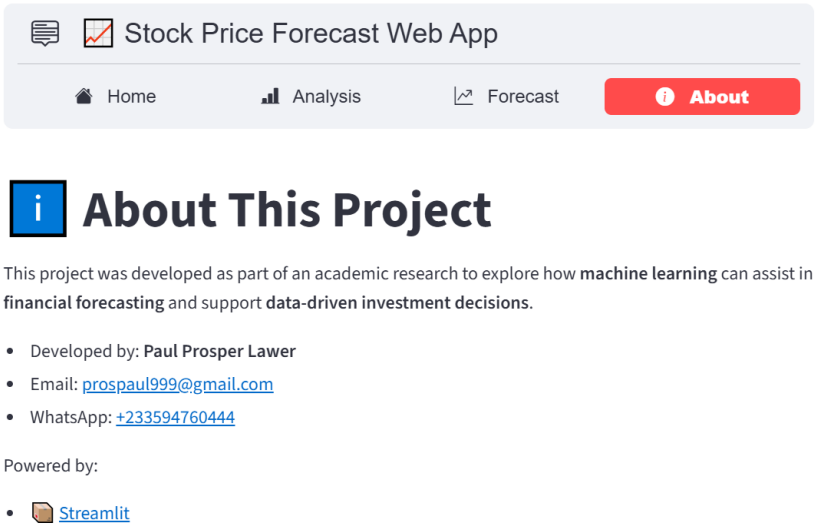
	Date	Close	High	Low	Open	Volume
		MTNOY	MTNOY	MTNOY	MTNOY	MTNOY
2630	2025-06-18 00:00:00	7.23	7.33	7.17	7.33	6000
2631	2025-06-20 00:00:00	7.11	7.17	7.05	7.17	10600
2632	2025-06-23 00:00:00	7.14	7.15	7.04	7.06	3600
2633	2025-06-24 00:00:00	7.47	7.5	7.35	7.41	10200
2634	2025-06-25 00:00:00	7.48	7.49	7.43	7.49	5800

Stock Opening & Closing Prices

4.3.1 User Interface (UI)

The user interface was built using Streamlit, chosen for its simplicity and seamless integration with Python. It provides:

- Drop-down selections for stock symbols
- Date input widgets for specifying forecasting periods
- Real-time display of charts and predictions



4.3.2 Backend Logic

The backend includes:

- Integration with the yfinance library to retrieve historical stock data.

- Use of Facebook's Prophet model for time-series forecasting.
- Data preprocessing to ensure clean input for the model.

4.3.3 Visualization

Plotly was used to generate:

- Line graphs for stock trends
- Candlestick charts for recent activity
- Forecast plots showing future trends up to 5 years

4.4 System Modules Integration

After individual modules were successfully implemented, integration focused on ensuring seamless data flow between components:

- Historical data fetched through yfinance is preprocessed and passed to the Prophet model.
- The model output is sent to visualization functions for rendering in the UI.
- The entire flow responds dynamically to user inputs (stock symbol, date range), updating charts in real time.

4.5 Testing

Testing was conducted continuously, following Agile principles, and structured into several phases to ensure comprehensive system validation.

4.5.1 Testing Plan

The testing plan included:

- Unit Testing: Verifying individual components (data fetch, forecasting model, chart generation).
- Integration Testing: Checking how components interact.
- System Testing: Ensuring overall functionality.

- User Testing: Gaining feedback from selected end-users.

4.5.2 Verification Testing

Verification focused on ensuring the application was built according to specifications:

- Confirmed correct integration of yfinance and Prophet.
- Ensured Streamlit rendered the UI accurately.
- Verified output consistency of prediction results under known input scenarios.

4.5.3 Validation Testing

Validation ensured the app met user requirements:

- Users could input dates and select stocks.
- Charts were accurate and understandable.
- Forecasts were generated and visualized clearly.
- Feedback from finance students and novice investors validated usability and clarity.

4.5.4 System Security Testing

Basic security testing was done to ensure:

- No unauthorized access to system files
- Inputs are sanitized to prevent injection or breakage
- External data sources (Yahoo Finance API) were handled securely with error checks

4.5.5 Recommendations Made by Testers

Testers provided the following recommendations:

- Add more companies to the stock list

- Implement a data caching mechanism to speed up repeated queries
- Include a summary section explaining prediction insights
- Improve error handling when API fails

4.5.6 Responses to Recommendations from Testing

In response:

- Additional companies were added based on relevance
- Caching was partially implemented for recent queries
- A simple prediction summary was added to the UI
- Try-except blocks and user-friendly error messages were incorporated

CHAPTER FIVE

5.1 Introduction

This chapter presents a summary of the research findings, draws conclusions based on the study's outcomes, identifies challenges encountered during implementation, outlines lessons learned, and provides recommendations for future work. The study focused on developing a web-based stock price prediction system using the Facebook Prophet model, designed to assist investors in making data-driven decisions.

5.2 Findings

- Functionality: The developed system can accurately forecast stock prices for selected companies using historical data retrieved via the yfinance API. Users can specify forecasting periods and visualize results with Plotly charts.
- Model Performance: The Facebook Prophet model proved effective for time-series forecasting with daily data. It performed well under normal market conditions and provided interpretable outputs.
- User Interface: Streamlit enabled the development of a user-friendly web interface. Users could interact with the app without needing technical expertise.
- Visualization: Interactive charts such as line plots, candlestick charts, and component forecasts enhanced user understanding of trends and model components.
- User Feedback: Initial user testing (with finance students and novice investors) confirmed that the application was easy to navigate, informative, and helpful for educational and investment purposes.

5.3 Conclusions

The research demonstrates that integrating machine learning models like Facebook Prophet into a web-based application can significantly enhance the ability of investors to forecast stock prices. The project achieved its goal of creating a functional, interactive, and accessible forecasting system. While the system is limited to selected companies and does not incorporate external market factors (e.g., news, earnings reports), it provides a solid foundation for time-series-based financial forecasting.

5.4 Challenges

- Data Quality and Availability: Stock data from Yahoo Finance occasionally had missing or inconsistent values, which required preprocessing.
- Model Limitations: Prophet assumes that future trends mirror past behaviors, which may not hold true in volatile or irregular markets.

- Deployment Issues: Hosting the application with external dependencies like yfinance and Prophet sometimes caused delays and required robust error handling.
- Limited Scope: The model did not incorporate non-price indicators such as sentiment analysis, financial ratios, or macroeconomic factors, which could improve accuracy.

5.5 Lessons Learnt

- Iterative Development Helps: Using Agile methodology allowed for continuous testing and user feedback, which improved the system's usability and performance.
- Data Preprocessing is Critical: Quality input data significantly influences model output, especially for time-series forecasting.
- Model Interpretability Matters: The choice of Prophet over more complex models (e.g., LSTM) balanced prediction accuracy and interpretability for users with non-technical backgrounds.
- User-Centered Design is Key: Simplicity in interface design enhanced accessibility, particularly for users unfamiliar with machine learning or programming.

5.6 Recommendations for Future Works

- Support for More Stocks
- Incorporation of External Variables
- Alternative Forecasting Models
- Mobile Compatibility
- Prediction Summary and Insights
- Real-time Data and Alerts

5.7 References

Ariyo, A. A., Adewumi, A. O., & Ayo, C. K. (2014). Stock price prediction using the ARIMA model. UKSim-AMSS 16th International Conference.

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PloS one, 12(7).

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time Series Analysis: Forecasting and Control. Wiley.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 270(2).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8).

Sean, J. T., & Letham, B. (2016). Forecasting at scale. The American Statistician, 72(1).

Taylor, S. J., & Letham, B. (2018). Forecasting at scale with Prophet. PeerJ Preprints.